Sospita License Protection

The QX Operating System

Security Target

Document number 1.2.2.2

Product version 3.2

11 September 2002

Public

Sospita
Head Office: Gjerdrums vei 10C • 0484 Oslo • Norway • Tel. +47 815 49 090 • Fax +47 815 49 091
R&D Office: Øksa 18 • P.O. Box 1417 • 4505 Mandal • Norway • Tel. +47 815 49 090 • Fax +47 815 49 091
U.S. Office: 6700 North Andrews Ave. • Ste. 400 • Ft. Lauderdale, FL 33309 • USA • Tel. +1 (954) 229 8283 • Fax +1 (954) 229 8301
www.sospita.com • Org. # NO 974341840 MVA

The information contained in this document is accurate to the best of Sospita's knowledge. However, Sospita disclaims any liability resulting from the use of this information and reserves the rights to make changes without notice.

Table of Contents

1 Introduction

This document describes the Security Target for the QX micro-controller operating system. It defines the security enforcing functions of the Target of Evaluation and the environments in which it operates.

This document should be read by people needing to understand the security functionality of the QX operating system.

2 Related documents and components

[CC] *Common Criteria for Information Technology Security Evaluation*, Version 2.1, August 1999 (aligned with ISO 15408).

[FIPS PUB 46-3] *DATA ENCRYPTION STANDARD (DES)*, Federal Information Processing Standards Publication 46-3, U.S. Department of Commerce / National Institute of Standards and Technology, Reaffirmed 1999 October 25.

[FIPS PUB 180-1] *SECURE HASH STANDARD*, Federal Information Processing Standards Publication 180-1, 1995 April 17.

[DEV_MANUAL] *Sospita License Protection, Developer's Manual*, August 24, 2002

[ADM_MANUAL] *Sospita License Protection, Administrator's Manual*, August 16, 2002

[SRS] *Sospita License Protection, Sospita Runtime System (includes the QX application program interface (API))*, version 3.2, August 2002.

3 Terminology

ACR	Access control rights – A field in a QX license and in a QX application
Authorised user	The holder of a QX token and the corresponding access information (PIN/PUK or password)
IT	Information Technology
License	A data structure that defines the rights to add protection and/or execute a particular QX application
Licensed application	A software application that is compiled for execution both in QX and on the host
Object	An entity within the TSC that contains or receives information and upon which subjects perform actions
PIN	Personal identification number
PP	Protection Profile - An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.
PUK	Personal unblocking key
QX	seQure eXecution - The token operating system made by Sospita and the TOE in this document
qxblock	A single application that is to be executed in QX
qxcode	A collection of one or more qxblocks for a licensed application
SDK	Sospita Development Kit (a component of SLP)
SF	Security Function - A part or parts of the TOE that have to be relied upon for enforcing a closely related subset of the rules from the TSP.
SFP	Security Function Policy
SLM	Sospita License Manager (a component of SLP)
SLP	Sospita License Protection
SRS	Sospita Runtime System (a component of SLP)
Subject	An entity within the TSC that causes operations to be performed
ST	Security Target - A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.
TOE	Target of Evaluation - An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.
Transport record	A collection of a license and information required for secure transfer of the license between two physically separated tokens
TSC	TOE Scope of Control - The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.
TSF	TOE Security Functions - A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.
TSF data	Data created by and for the TOE, that might affect the operation of the TOE. Here, TSF data is all data created solely by QX. Currently, this is only session key used in license transfer.
TSP	TOE Security Policy - A set of rules that regulate how assets are managed, protected and distributed within a TOE.
User data	Data created by and for the user, that does not affect the operation of the TSF. Here, the user data is all data created within QX, i.e. qxblocks and all attributes of licenses.

4 Introduction to the Security Target

4.1 Security Target Identification

Document title:

Sospita License Protection, The QX Operating System, Security Target, Document number 1.2.2.2, Product version 3.2, 11 September 2002.

Assurance Level: EAL3

Author: Pål Berg (pal.berg@Sospita.com)

4.1.1 TOE Identification

Unique ATR[1] (Answer To Reset) (hex):

Philips P16Wx064: 3B 7B XX 00 00 A0 00 00 01 06 51 58 32 03 01 11

Atmel AT90SC6464C: 3B 7B XX 00 00 A0 00 00 01 06 51 58 32 03 01 05

USB identification[2, 3] (hex):

Philips P16Wx064: 0FE6 0001

Atmel AT90SC6464C-USB: 0FE6 0002

4.2 Security Target Overview

This Security Target is written for the QX operating system, designed for hardware tokens. QX is a micro-controller operating system and it is an important part of Sospita License Protection (SLP). SLP provides license controlled execution of software applications. Encrypted QX applications can be uploaded dynamically, decrypted and executed under the control of a license, and licenses can be transferred securely between token.

4.3 CC Conformance Claim

This TOE makes the following conformance claims with respect to [CC]:

- Part 2 conformant
- Part 3 conformant
- EAL3 conformant

[1] 3B 7B XX 00 00: ISO header (XX depends on the communication bandwidth and is editable)
A0 00 00 01 06: Sospita ISO identification number
51 58: String: "QX"
32 03: QX build number
01: Product information byte
Last byte: Platform status byte, 11: Philips P16WX064, 05: Atmel AT6464C

[2] 0FE6 is Sospita's identification number, 0001: Philips P16WX064, 0002: Atmel AT90SC6464C-USB

[3] Note that the version (build) number is found by the QXGetTokenVersion API call.

5 TOE Description

5.1 Overview

The TOE of this Security Target is the QX micro-controller operating system made by Sospita. QX forms a vital part of the security in Sospita License Protection (SLP). SLP is a product that provides protection of software applications against unauthorized usage (software piracy). The main feature of SLP is license controlled execution of QX applications, where the QX applications are vital for the software application executing on the host computer. Using the Sospita Development Kit (SDK), which is outside of the scope of the evaluation, a developer splits software applications into elements that after compilation execute either on the token or on the host. Remark that this can have other utilization areas than just piracy prevention, but can be used for generic token applications as well. The elements to be executed under QX are called qxblocks, and a collection of qxblocks for one application forms a qxcode. These are commonly referred to as QX applications. This compilation is sketched in fig. 1.

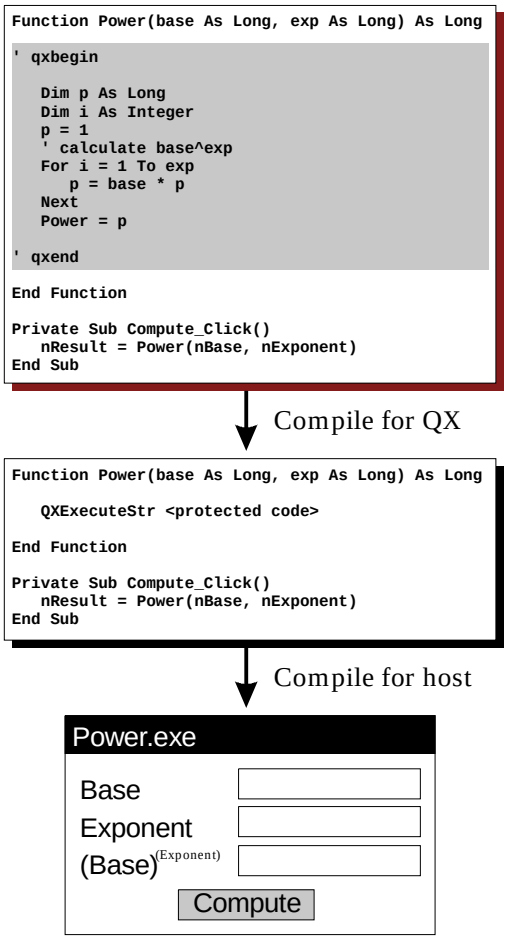

Figure 1 – Generation of a QX application. The example uses a function for calculating the power of a number. The code to be protected (gray) is identified by the use of the tags "qxbegin" and "qxend".

During runtime of the software application, the Sospita Runtime System (SRS) uploads (if not already done) the protected QX applications to QX. In QX, the protected application is decrypted and executed, and the result is returned to the host. Following fig. 1, the next figure shows the execution phase the power function.

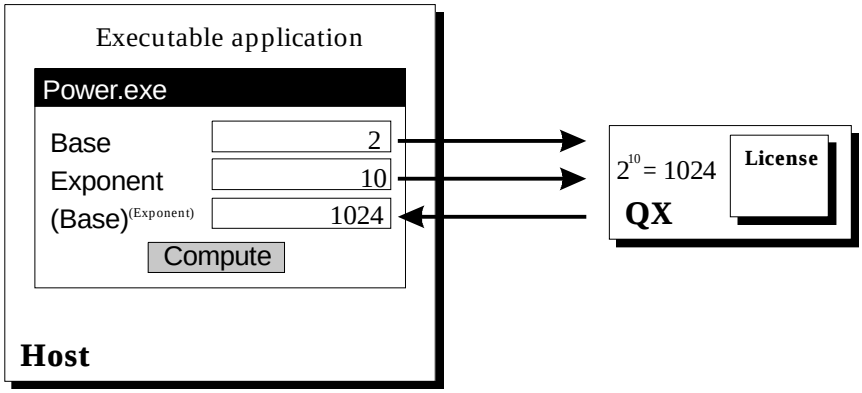

Figure 2 – Execution of the power function generated in fig. 1. The input numbers to QX is 2 and 10, and QX then calculates 2^{10}, which is 1024. This number is then returned to the host.

The important feature here is the license which must be present in QX in order to execute the QX application. A license for the end-user will be a non-writeable data element which can be identified by a identification number. In addition, a license can hold constraints for determining the validity of the license. The licensing model opens for a software distribution model where the software can be distributed freely and without the risk of piracy, but where the execution itself is controlled. This also means that QX must be resident in a tamper resistant environment, and for that reason micro-controllers typically used for smart card applications are employed. The QX life-cycle is source code development and compilation for the specific hardware platform. If the hardware is ROM based, then the next step is mask production, else if QX shall exist in flash memory, the compiled code can be uploaded directly. The underlying hardware of the token is not in the scope of the evaluation.

QX has three principal security objectives:

- All execution of QX applications must be controlled by a license;
- Any security relevant information resident in QX, including the protected parts of an application and license data must never appear in clear on the host;
- All QX applications are considered mutually hostile, and one application running in QX must not be permitted to interfere with another running in QX;

The QX operating system has two components: a Security Manager (SM) and a Virtual Machine (VM). The Virtual Machine controls the execution of QX applications (including memory management and connections), whilst the Security Manager performs the following operations:

- Control of all input and output to and from the host;
- Enforcement of license restrictions;
- Performs cryptographic operations.

As already stated, SLP uses licenses to control the execution of software applications. This means that it must be possible to distribute licenses to end-users. The software vendor obtains a (partly) writeable master license resident on a QX token from a master license generation facility, and uses the master license in protection of parts of the software application after the license has been initialized with cryptographic data and constraints. The license is then converted to a non-writeable end-user. For more complex systems, the license can also be converted to an end-user license with splitting and merging properties. This is then called a server license. The non-writeable attributes of a master license include the license id, the number of instances and reference to a root transport license.

To ensure secure transfer of licenses between QX tokens, transport licenses will be used. Initially, all tokens must have a root transport license, and it is possible for vendors to create their own transport licenses. The license life cycle is given in fig. 3. Remark here that the next time the vendor or end-user needs another license, it is not necessary with a new token from Sospita as one token can hold a number of licenses.

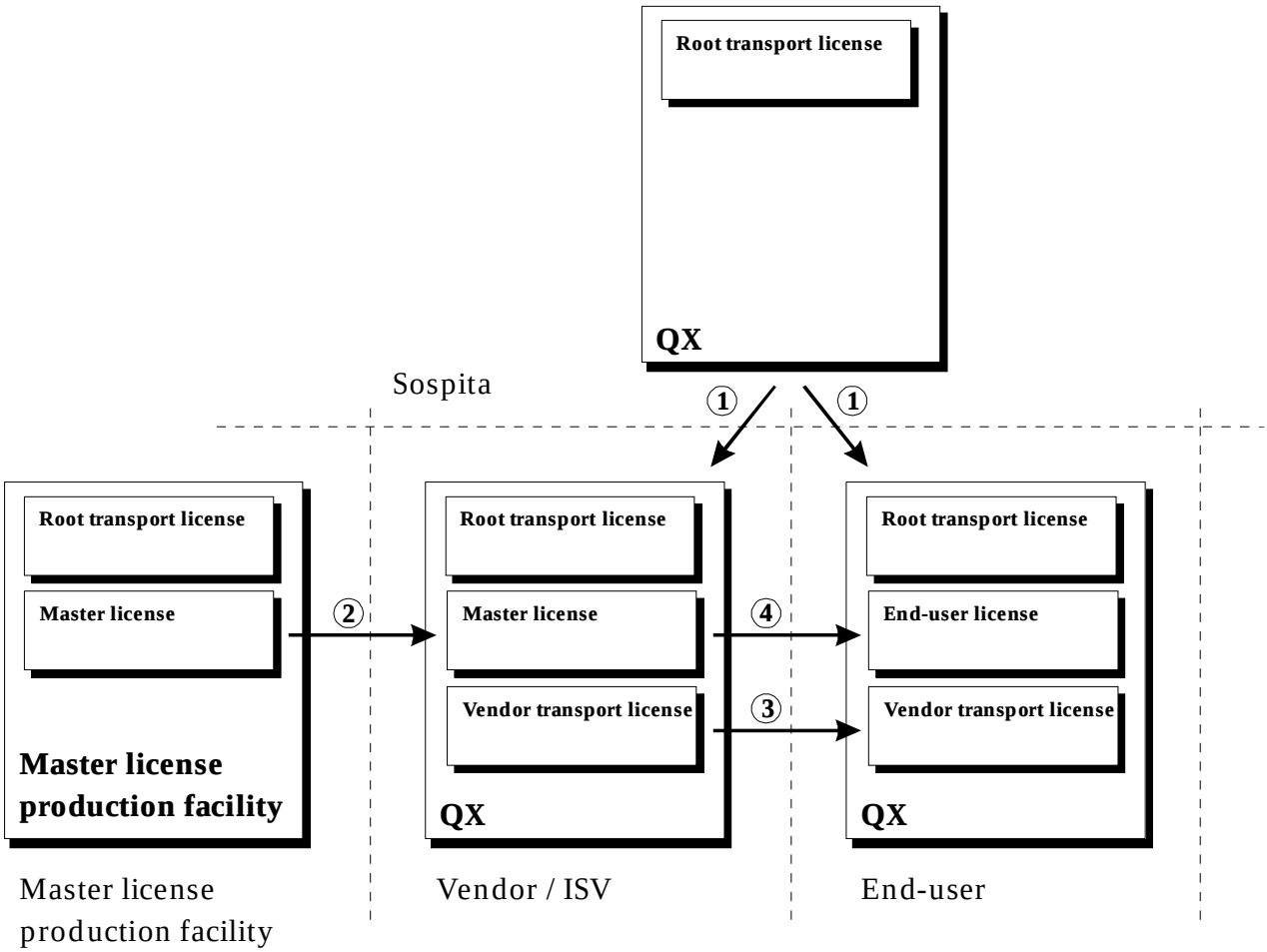

Figure 3 – License life cycle. When a QX token has a root transport license it can be distributed to software vendors and end-users. The vendor then obtains a master license from a master license production facility and creates a transport license. The end-user can then securely transfer both licenses (the vendor transport license must be transferred first).

A license has several attributes which are important for the security. The most security relevant attribute is the cryptographic information used in encryption of application code. A unique license identification number (ID) and a parameter stating the number of license copies or instances, is also important for the licensing concept.

Licenses can be locked by the use of a PIN/PUK or password. The qualities of the PIN/PUK and password are checked by the Sospita Runtime System (SRS) [SRS], thus the SRS is a required component in this evaluation.

5.2 TOE Hardware Requirements

The TOE runs on a hardware token that is both a secure storage device for licenses and a secure environment for executing the protected part of a software environment. For the purposes of the evaluation the hardware token shall be the **Philips P16WX064[4], Atmel AT90SC6464C** and **Atmel AT90SC6464C-USB** micro-controllers.

5.3 Scope of Evaluation

The scope of the TOE is limited to the QX operating system. The following components of Sospita License Protection are outside of the scope of the evaluation, but relevant for secure startup and environment:

- Sospita Development Kit [DEV_MANUAL];
- Sospita License Manager [ADM_MANUAL];
- Sospita Runtime System [SRS].

[4] Currently, under evaluation to Common Criteria EAL5+

6 Security Environment

6.1 Introduction

This section provides the statement of the TOE security environment, which identifies and explains all:

- Known and presumed threats countered by either the TOE or by the security environment;
- Organizational security policies with which the TOE must comply;
- Assumptions about the secure usage of the TOE, including physical, personnel and connectivity aspects.

6.2 Threats

This section identifies the threats to the IT assets against which protection is required by the TOE or by the security environment.

6.2.1 Assets

The primary assets requiring protection are the licensed software applications, which must be protected from unauthorized usage.

It is also useful to consider the following secondary assets related to the specific form of implementation described in this security target:

- License – a data set containing attributes that define the usage constraints of a protected application, and the key used for encrypting, and during execution, decrypting, a software application. This data is stored on a QX token.

- Data held in Secure Token processor, storage and memory – physical aspects of token protection are addressed by the TOE environment, but the TOE itself must provide protection against attempts to retrieve information (e.g. decrypted application segments) from the token during operation.

6.2.2 Threat Agent

The threat agent is a user of the protected application, with access to the installation software, installed software and the token. The user is assumed to be an attacker with access to disassembler tools, and knowledge of the design and implementation of the TOE. In addition, the threat agent has access to the SDK for designing and protecting own code, by which he can systematically observe the behavior of the protected code.

6.2.3 Threats Countered by the TOE

The following specific threats are countered:

T.MODIFY	An attacker may successfully modify a licensed software application or components of a licensed software application, resulting in execution of an application that they are not licensed to use.
T.CONFIDENTIAL	An attacker may successfully get security sensitive attributes of a license.
T.LICENSE	An attacker may successfully modify non-writable attributes of a license, causing the license to be illegally duplicated or extending the original limitations.
T.TIME	An attacker may successfully execute a software application outside of the time limits imposed by the license without tampering with the license itself.
T.NUMBER	An attacker may successfully execute a software application more times than allowed by the limits imposed by the license without tampering with the license itself.
T.ACR	An attacker may successfully execute a software application with different ACR than allowed by the limits imposed by the license without tampering with the license itself.
T.INTERFERE	Information may leak between QX applications or between licenses and QX applications, allowing an attacker to breach license protection.
T.COPY	An attacker may successfully illegally duplicate a license.
T.MASTER	An attacker may obtain a master license with an id reserved for somebody else. This impersonation can cause application errors if a user tries to execute an application with the attacker's license or simplify unauthorized duplication if the attacker has access to cryptographic information used in original master license.
T.USE	A non-authorized user may break the lock protection of a PIN/PUK or password locked license or token, and perform unauthorized usage.
T.STRESS	An attacker can stress the physical limitations on write cycles of EEPROM and get a modification of bits in licenses.

6.2.4 Threats Countered by the Operating Environment

The following threat is required to be countered by technical and/or non-technical measures in the IT environment:

T.SECURE	A sophisticated attacker possessing the necessary skills and appropriate hardware and software tools may perform hardware attacks in an attempt to modify or read security sensitive data in the

TOE.

6.3 Organizational Security Policies

There are no organizational security policies or rules with which the TOE must comply.

6.4 Assumptions

Conditions regarding the security aspects of the environment in which the TOE is intended to be used are fully covered by the environmental security objectives that are described in chapter 7.2. There are no further assumptions that need to be described in this chapter.

7 Security Objectives

7.1 TOE Security Objectives

7.1.1 IT Security Objectives

The specific IT security objectives are as follows:

O.EXECUTE	One or more licenses control the execution of QX applications.
O.CONFIDENTIAL	An end-user must not be able to view security sensitive attributes of a license stored on a token, but may be able to view those that are not security sensitive.
O.INTEGRITY	A user must not be able to use modified QX applications by which he/she is not the legal owner of, i.e. possesses a corresponding master license.
O.NONINTERFERE	The execution of a QX application on a token must not interfere with any other QX application or license.
O.PROTECT	QX applications must not be readable from the host computer.
O.TIMELIMIT	Execution of a QX application must be permitted only within the time constraints of a license stored on a token.
O.NUMLIMIT	The number of executions of a QX application must be controlled in accordance with a counter of license stored on a token.
O.ACR	The execution of QX applications must be controlled in accordance with the ACR field stored in licenses.
O.TRANSPORT	The ability to securely (integrity and confidentiality protection) transfer a license between tokens must be provided.
O.COPY	It must be possible to control the ability to propagate a license from one token to other tokens.
O.USE	A user wanting to use a PIN/PUK or password locked license or token must provide a PIN/PUK or password in order to be authorized.
O.LICENSE	Modification of attributes of a license must be controlled.
O.STRESS	Mechanisms for detecting hardware faults because of imposed stress must be provided by the TOE.
O.ID	Mechanisms for providing a proof of ownership for a specific master license must exist.

7.2 Environment Security Objectives

7.2.1 IT Environment Security Objectives

The following IT security objectives are to be satisfied by the environment:

OE.INSTALL	The TOE must be installed on a secure hardware token that provides resistance against physical attacks appropriate to the threat environment.
OE.SRS	The QX application program interface [SRS] will provide security metrics for the authorizing mechanisms (i.e. PIN/PUK and password).
OE.MASTER	The ability to generate master licenses must be limited to a master license generation facility.

7.2.2 Non-IT Environment Security Objectives

Non-IT environment security objectives are to be satisfied without imposing technical requirements on the TOE. That is, they will not require the implementation of functions in the TOE hardware and/or software. Thus they will be implemented largely through procedural or administrative measures.

OE.APPLIC	Protected software applications and the corresponding cryptographic information must be generated and stored in a secure manner, such that no sensitive data is disclosed or possible to modify for unauthorized persons.

8 IT Security Requirements

8.1 TOE Security Functional Requirements

The TOE security functional requirements are presented in this section. The following table summarizes those security requirements. Completed operations are shown in bold. Iteration is indicated by use of (*n*) following the component designator, where *n* is the number of the iteration.

Functional Components	
FCS_CKM.1	Cryptographic key generation
FCS_COP.1 (1)	Cryptographic operation
FCS_COP.1 (2)	Cryptographic operation
FCS_COP.1 (3)	Cryptographic operation
FDP_ACC.1 (1)	Subset access control
FDP_ACC.1 (2)	Subset access control
FDP_ACF.1 (1)	Security attribute based access control
FDP_ACF.1 (2)	Security attribute based access control
FDP_DAU.1	Basic data authentication
FDP_ITC.2 (1)	Import of user data with security attributes
FDP_ITC.2 (2)	Import of user data with security attributes
FDP_RIP.1	Subset residual information protection
FDP_SDI.1	Stored data integrity monitoring
FDP_UCT.1 (1)	Basic data exchange confidentiality
FDP_UCT.1 (2)	Basic data exchange confidentiality
FDP_UIT.1 (1)	Data exchange integrity
FDP_UIT.1 (2)	Data exchange integrity
FIA_AFL.1 (1)	Authentication failures
FIA_AFL.1 (2)	Authentication failures
FIA_AFL.1 (3)	Authentication failures
FIA_UAU.1	Timing of authentication
FIA_UAU.6	Re-authenticating
FPT_RPL.1	Replay detection
FPT_RVM.1	Non-bypassability of the TSP
FPT_SEP.1	TSF domain separation

Table 8-1 Functional requirements

FCS_CKM.1 **Cryptographic key generation**

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [**triple DES**] and specified cryptographic key sizes [**112 bits**] that meet the following: [**no standards**].

FCS_COP.1 (1)	**Cryptographic operation**
FCS_COP.1.1(1)	The TSF shall perform [**encryption and decryption of QX applications**] in accordance with a specified cryptographic algorithm [**tripleDES**] and cryptographic key sizes [**112 bits**] that meet the following: [**FIPS PUB 46-3**].
FCS_COP.1 (2)	**Cryptographic operation**
FCS_COP.1.1(2)	The TSF shall perform [**encryption and decryption of security fields in a transport record**] in accordance with a specified cryptographic algorithm [**triple-DES**] and cryptographic key sizes [**112 bits**] that meet the following: [**FIPS PUB 46-3**].
FCS_COP.1 (3)	**Cryptographic operation**
FCS_COP.1.1(3)	The TSF shall perform [**hash generation**] in accordance with a specified cryptographic algorithm [**SHA-1**] and cryptographic key sizes [**160 bits**[5]] that meet the following: [**FIPS PUB 180-1**].
FDP_ACC.1 (1)	**Subset access control**
FDP_ACC.1.1 (1)	The TSF shall enforce the **[license access control SFP]** on: **a)** [**The following subjects:** • **User** **b) The following objects:** • **Master license** • **Server license** • **End-user license** • **Transport license** **c) The following operations:** • **View a license** • **Generate a request for a master license** • **Generate a transport license** • **Move a license between tokens** • **Derive a specific number of end-user licenses** • **Merge end-user and server licenses** • **Backup writable licenses** • **Edit writable license attributes** • **Lock and unlock a license** • **Delete a license**].

[5] In the case of a hash function the cryptographic key size refers to the length of the hash.

FDP_ACC.1 (2)	**Subset access control**
FDP_ACC.1.1 (2)	The TSF shall enforce the **[QX application access control SFP]** on **a)** [**The following subjects:** • **User** **b) The following objects:** • **QX application** **c) The following operations:** • **Upload a QX application** • **Encrypt and decrypt a QX application** • **Execute a QX application**].

FDP_ACF.1 (1)	**Security attribute based access control**
FDP_ACF.1.1 (1)	The TSF shall enforce the [**license access control SFP**] to objects based on • [**The type of license** • **The license attributes** • **The type of operation**].

FDP_ACF.1.2 (1) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- [**A user shall be able to view only those attributes of an end-user license that are not security sensitive, i.e. no cryptographic keys and initialization vectors, and lock parameters.**
- **A user shall have the ability to generate a request for a master license.**
- **A user with a master license shall have the ability to generate a transport license with id one number bigger than the master license id.**
- **When a transport license and the parent master license are present on the same token, attributes of the transport license shall be writable. Otherwise, the license shall be non-writable.**
- **A user of a token containing a transport license shall have the ability to move licenses to other tokens that have a transport license with the same license id and same cryptographic parameters, and it shall no be possible to replay/double spend the license being transported**
- **A user of a token containing a specific number of master licenses shall only have the ability to derive a specific number of end-user or server licenses.**
- **A user of a token containing a specific number of server licenses with a specific constraint shall only have the ability to derive a specific number of end-user with limitations as imposed by the constraints.**
- **It shall be possible to merge end-user licenses into server licenses if the specific constraints allow for this operation.**
- **A user of a token containing a master license shall have the ability to write to relevant attributes of the license.**
- **A user of a token containing a master or an editable transport license shall have the ability to link this license to a transport license that is no further up in a hierarchy[6] than an ancestor of a controlled transport license.**
- **A user shall have the ability to delete a license that is at a terminal node of a hierarchy.**

[6] The terminology of a hierarchy is *root* at the top. A *child* license contains a link to a *parent* license. A hierarchy with two or more licenses will have *descendant(s)* and *ancestor(s)*. The licenses at the end of a hierarchy, which is not the root, are called *terminal nodes* or licenses.

- **A user of a token shall have the ability to backup a master license whereby the backed up license contains a zero instance counter.**
- **A license issuer shall have the ability to decide if the issued non-writeable transport and end-user license shall be moveable.**
- **A user of a token shall have the ability to backup a writable transport license.**
- **The non-writable attributes of a license shall not be modified by anyone.**
- **A user can lock a license with a pin or password if the license allows for this.**
- **A pin-blocked lock of a license can be unblocked by providing a puk.**].

FDP_ACF.1.3 (1) The TSF shall explicitly authorise access of subjects to objects based on the following additional rules:

- [**A user wanting to use a locked license must provide a pin or password in order to be authorized.**].

FDP_ACF.1.4 (1) The TSF shall explicitly deny access of subjects to objects based on the [**none**].

FDP_ACF.1 (2) Security attribute based access control

FDP_ACF.1.1 (2) The TSF shall enforce the [**QX application access control SFP**] to objects based on

- [**The attributes of a license**
- **The attributes of a QX application**
- **Date and time**].

FDP_ACF.1.2 (2) The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

- [**The TSF shall encrypt an unprotected QX application only if the token contains a master license that is valid[7].**
- **The TSF shall upload, decrypt and execute a QX application only if the token contains an end-user or master license that is valid.**
- **The TSF shall upload, decrypt and execute a QX application only if the token contains an end-user or master license that is not in use by another QX application.**
- **The TSF shall prevent a user from reading a decrypted QX application.**].

[7] Validity of a license is decided from limitations, access control rights and cryptographic parameters.

FDP_ACF.1.3 (2)	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: • [**Execution of a QX application shall be denied if it depends on a license that is protected by a pin or password, and the correct one is not supplied**].
FDP_ACF.1.4 (2)	The TSF shall explicitly deny access of subjects to objects based on the [**none**]**.**
FDP_DAU.1	**Basic data authentication**
FDP_DAU.1.1	The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [**requests for a master license with a given id**].
FDP_DAU.1.2	The TSF shall provide [**none**] with the ability to verify evidence of the validity of the indicated information.
FDP_ITC.2 (1)	**Import of user data with security attributes**
FDP_ITC.2.1 (1)	The TSF shall enforce the [**license access control SFP**] when importing user data, controlled under the SFP, from outside of the TSC.
FDP_ITC.2.2 (1)	The TSF shall use the security attributes associated with the imported user data.
FDP_ITC.2.3 (1)	The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the data received.
FDP_ITC.2.4 (1)	The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.
FDP_ITC.2.5 (1)	The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: [**None**].
FDP_ITC.2 (2)	**Import of user data with security attributes**
FDP_ITC.2.1 (2)	The TSF shall enforce the [**QX application access control SFP**] when importing user data, controlled under the SFP, from outside of the TSC.
FDP_ITC.2.2 (2)	The TSF shall use the security attributes associated with the imported user data.
FDP_ITC.2.3 (2)	The TSF shall ensure that the protocol used provides for the unambiguous association between the security attributes and the data received.
FDP_ITC.2.4 (2)	The TSF shall ensure that interpretation of the security attributes of the imported user data is as intended by the source of the user data.

FDP_ITC.2.5 (2)	The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TSC: [**None**].
FDP_RIP.1	**Subset residual information protection**
FDP_RIP.1.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the [**allocation of the resource to**] the following objects: [**memory locations used by** • **a QX application;** • **a license**]**.**
FDP_SDI.1	**Stored data integrity monitoring**
FDP_SDI.1.1	The TSF shall monitor user data stored within the TSC for [**integrity errors**] on all objects, based on the following attributes: [**message digests of licenses and checksums of QX application's memory range**].
FDP_UCT.1 (1)	**Basic data exchange confidentiality**
FDP_UCT.1.1 (1)	The TSF shall enforce the [**license access control SFP**] to be able to [**transmit**] objects in a manner protected from unauthorized disclosure.
FDP_UCT.1 (2)	**Basic data exchange confidentiality**
FDP_UCT.1.1 (2)	The TSF shall enforce the [**QX application access control SFP**] to be able to [**transmit**] objects in a manner protected from unauthorized disclosure.
FDP_UIT.1 (1)	**Data exchange integrity**
FDP_UIT.1.1 (1)	The TSF shall enforce the [**license access control SFP**] to be able to [**transmit**] user data in a manner protected from [**modification**] errors.
FDP_UIT.1.2 (1)	The TSF shall be able to determine on receipt of user data, whether [**modification**] has occurred.
FDP_UIT.1 (2)	**Data exchange integrity**
FDP_UIT.1.1 (2)	The TSF shall enforce the [**QX application access control SFP**] to be able to [**transmit**] user data in a manner protected from [**modification**] errors.
FDP_UIT.1.2 (2)	The TSF shall be able to determine on receipt of user data, whether [**modification**] has occurred.

FIA_AFL.1 (1)	**Authentication failures**
FIA_AFL.1.1 (1)	The TSF shall detect when [**three**] unsuccessful authentication attempts occur related to [**unlock a PIN-locked license or token**].
FIA_AFL.1.2 (1)	When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [**block the lock of the license or token such that it can only be unblocked by a PUK**].
FIA_AFL.1 (2)	**Authentication failures**
FIA_AFL.1.1 (2)	The TSF shall detect when [**six**] unsuccessful authentication attempts occur related to [**unlock a PIN-locked license or token where the license or token only can be unlocked by the use of a PUK**].
FIA_AFL.1.2 (2)	When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [**permanently block the license or token**].
FIA_AFL.1 (3)	**Authentication failures**
FIA_AFL.1.1 (3)	The TSF shall detect when [**ten**] unsuccessful authentication attempts occur related to [**unlock a password-locked license or token**].
FIA_AFL.1.2 (3)	When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [**permanently block the license or token**].
FIA_UAU.1	**Timing of authentication**[8]
FIA_UAU.1.1	The TSF shall allow [**any licensed activity not protected by a lock**] on behalf of the user to be performed before the user is authenticated.
FIA_UAU.1.2	The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.
FIA_UAU.6	**Re-authenticating**
FIA_UAU.6.1	The TSF shall re-authenticate the user under the conditions [**when a token is re-connected to the host**].

[8] This SFRs reflects the requirement where a license is locked.

FPT_RPL.1	**Replay detection**
FPT_RPL.1.1	The TSF shall detect replay for the following entities: [**a license in transfer that is uploaded more than once to the destination token**] and [**a license in transfer that is uploaded to another token than the specified destination token**].
FPT_RPL.1.2	The TSF shall perform [**prevention of this operations**] when replay is detected.
FPT_RVM.1	**Non-bypassability of the TSP**
FPT_RVM.1.1	The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.
FPT_SEP.1	**TSF domain separation**
FPT_SEP1.1	The TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.
FPT_SEP.1.2	The TSF shall enforce separation between the security domains of subjects in the TSC.

8.2 IT Environment Security Functional Requirements

The security functional requirements for the IT environment are presented in this section. The following table identifies those security requirements.

Functional Components	
FDP_DAU.1	Basic data authentication
FIA_SOS.1	Verification of secrets
FPT_PHP.1	Passive detection of physical attack
FPT_PHP.3	Resistance to physical attack
FPT_STM.1	Reliable Time Stamps

Table 8-2 Environment functional requirements

FDP_DAU.1	**Basic data authentication**
FDP_DAU.1.1	The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [**none**].
FDP_DAU.1.2	The TSF shall provide [**a master license generation facility**] with the ability to verify evidence of the validity of the indicated information.

FIA_SOS.1	**Verification of secrets**
FIA_SOS.1.1	The TSF shall provide a mechanism to verify that secrets meet [**a defined metric of passwords and PIN/PUK**].
FPT_PHP.1	**Passive detection of physical attack**
FPT_PHP.1.1	The TSF shall provide unambiguous detection of physical tampering that might compromise the TSF.
FPT_PHP.1.2	The TSF shall provide the capability to determine whether physical tampering with the TSF's devices or TSF's elements has occurred.
FPT_PHP.3	**Resistance to physical attack**
FPT_PHP.3.1	The TSF shall resist [**physical tampering**] to the [**token**] by responding automatically such that the TSP is not violated.
FPT_STM.1	**Reliable time stamps**
FPT_STM.1.1	The TSF shall be able to provide reliable time stamps for use by the TOE.

8.3 TOE Security Assurance Requirements

The security assurance requirements are taken from Part 3 of the CC and are those that comprise the EAL3 assurance package. The assurance components for EAL3 are identified in the following table.

Assurance Class	Assurance Components	
Configuration management	ACM_CAP.3	Authorization Controls
	ACM_SCP.1	TOE CM coverage
Delivery and operation	ADO_DEL.1	Delivery Procedures
	ADO_IGS.1	Installation, generation and start-up procedures
Development	ADV_FSP.1	Informal Functional Specification
	ADV_HLD.2	Security enforcing high-level design
	ADV_RCR.1	Informal correspondence demonstration
Guidance documents	AGD_ADM.1	Administrator guidance
	AGD_USR.1	User guidance
Life cycle support	ALC_DVS.1	Identification of security measures
Tests	ATE_COV.2	Analysis of coverage
	ATE_DPT.1	Testing: high-level design
	ATE_FUN.1	Functional testing
	ATE_IND.2	Independent testing – sample

Assurance Class	Assurance Components	
Vulnerability assessment	AVA_MSU.1	Examination of Guidance
	AVA_SOF.1	Strength of TOE security function evaluation
	AVA_VLA.1	Developer vulnerability analysis

Table 8-3 Assurance requirements: EAL3

Further information on these assurance components can be found in [CC] Part 3.

8.4 Strength of Function Claim

A Strength of Function (SoF) claim of SoF-HIGH is made for the TOE.

9 TOE Summary Specification

9.1 TOE Security Functions

This section describes the security functions provided by the TOE to meet the security functional requirements specified for the QX Operating System in section 8.1.

F1 The execution of QX application in a token is controlled by end-user, server or master licenses.

F2 It is possible to convert a master license to an end-user or server license. The license instance counter, NoOfLicenses, determined by the master license production facility, in the master license indicates the maximum number of end-user or server licenses that can be generated from this specific master license.

F3 A user is able to generate transport licenses on a token containing a master license, and in which the license id is one number bigger than the id of the master license and the transport license shall only be editable when this master license is present on the same token as the transport license.

F4 Means for transporting licenses between two tokens exists, and for this function a transport license must be used. This operation shall only be possible if both tokens contain transport licenses with the same id and cryptographic information, and the license issuer has allowed for transfer of the license. The transport license is used in confidentiality and integrity protection of the license in transport.

F5 The integrity of a license being transported or backed up is protected through use of an encrypted hash of the license.

F6 The license instance counter, NumberOfLicenses, of a specific master, server or end-user license is decremented before a license transport record containing this license is transmitted from the token. The decrease indicates the number of instances being transferred.

F7 It is not possible to upload a license transport record twice to a destination token if the first attempt was successful.

F8 It is not possible to upload a transport license record to a token that is not the given destination token.

F9 A request for a master license must include a proof of ownership of license id if the request is for more copies of an already allocated license id.

F10 A user is able to backup and restore master and writeable transport licenses. For master licenses, the backed up license record contains a zero instance counter. Backup and restore do not follow the rules of functions F7 and F8.

F11 A user is able to view license attributes that are not security sensitive[9].

F12 A user is able to delete a license which is at the terminal node of a hierarchy.

[9] The security sensitive attributes of a license are the secret cryptographic key and the initialization vector, in addition to lock specifications. Remark that master licenses can have fast encryption enabled, meaning that the cryptographic information is copied to the host., but this option is disabled for end-user, server and transport licenses.

F13 A license is able to control the execution of exactly one QX application at a time.

F14 One QX application on a token is not able to interfere with the execution path of any other QX application on the token, nor use memory areas used for licenses. When a memory area used by one QX application is allocated to another, this memory is cleared before the allocation.

F15 The uploading of QX application to a token is controlled by end-user, server or master licenses.

F16 The following attributes of licenses in transfer are protected from disclosure:

i The symmetric encryption key

ii The initialization vector

iii Lock specifications

F17 End-user and server licenses stored on QX tokens are non-writeable.

F18 A QX application stored outside a token is protected from unauthorized disclosure and modification. It is further not possible to view a decrypted QX application stored in a token.

F19 A session key[10] is used for encryption of security attributes in a transport record. The session key is further encrypted with a key in a transport license and included in the transport record.

F20 A user can edit writeable fields in a master and writeable transport license.

F21 A lock can be assigned to a license or token to ensure that only authenticated users can use this license or token. The user must provide a PIN or password in order to unlock a license or token and the unlocked modus is only active in a continuous session[11].

F22 After 3 successive authentication failures a PIN-locked license or token will be blocked until a correct PUK is provided. Wrong PUK can be given 6 times. After 6 successive PUK failures, the license or token will be permanently blocked.

F23 After 10 successive authentication failures a password-locked license or token will be permanently blocked.

F24 A user of a token containing a master or an writeable transport license shall have the ability to link these license to a transport license that is no further up in a hierarchy than an ancestor of a writeable transport license.

F25 The TOE shall detect write failures in EEPROM when licenses are uploaded[12].

F26 The TOE shall detect if integrity errors of licenses and memory range of QX applications stored on tokens occurs.

F27 A server license can be split into a given number of end-user licenses, limited by the NoOfLicenses attribute of the server license, and end-user licenses can be merged into a server license if the given constraint type allows for this.

9.2 Assurance Measures

Deliverables will be produced to comply with the Common Criteria security assurance requirements for EAL3.

[10] A random number generator in the hardware platform will be used in this operation.
[11] The token is connected to the host.
[12] Happens when the license transport record is moved from token RAM to EEPROM.

10 Protection Profiles Claims

There are no Protection Profile Claims.

11 Rationale

11.1 Introduction

This section identifies the rationale for the adequacy of the security functional requirements and the security assurance requirements in addressing the threats and meeting the objectives of the TOE.

11.2 Security Objectives for the TOE and Environment Rationale

The following table demonstrates how the objectives of the TOE and the TOE environment counter the threats, policies and assumptions identified in Section 3.2.1.

OBJECTIVES	Threats											
	T.MODIFY	T.COPY	T.LICENSE	T.CONFIDENTIAL	T.TIME	T.NUMBER	T.ACR	T.INTERFERE	T.MASTER	T.USE	T.STRESS	T.SECURE
O.EXECUTE	✓				✓	✓	✓					
O.CONFIDENTIAL	✓			✓						✓		
O.INTEGRITY	✓					✓	✓					
O.NONINTERFERE								✓				
O.LICENSE		✓	✓						✓			
O.PROTECT	✓											
O.TIMELIMIT					✓							
O.NUMLIMIT						✓						
O.ACR							✓					
O.TRANSPORT		✓	✓	✓					✓	✓		
O.COPY		✓										
O.USE										✓		
O.STRESS			✓								✓	
O.ID		✓							✓			
OE.APPLIC	✓	✓		✓					✓			
OE.INSTALL	✓	✓	✓	✓		✓	✓		✓	✓		✓
OE.MASTER		✓							✓			
OE.SRS										✓		

Table 11-1 Objectives rationale

As can be seen from the table above, all threats and assumptions are met by at least one objective of, either the TOE or environment, as applicable. The threats and assumptions countered by the TOE are discussed in the subsections below.

T.MODIFY deals with the general threat of unauthorized access to software applications. The TOE addresses this threat by controlling the ability to execute parts of the application (O.EXECUTE), and by protecting security sensitive attributes of a license from being viewed (O.CONFIDENTIAL). When the QX applications are resident on the host, they are encrypted and integrity protected (O.INTEGRITY). During execution, they are uploaded to a secure token for decryption and integrity verification. The protected parts of the application cannot be read from the host computer (O.PROTECT). Within the TOE environment it is necessary to use the Sospita Development Kit to protect the software in an appropriate secure manner (OE.APPLIC). The TOE hardware must also be protected against physical attack (OE.INSTALL).

T.COPY identifies a specific subset of the threat related to copying of licenses. This specific threat is addressed by O.COPY and O.TRANSPORT. This objective is further supported by the security by the vendor (OE.APPLIC) and the underlying tamper resistance of the hardware (OE.INSTALL). It shall also not be able to modify non-writeable license parameters (O.LICENSE), like the attribute indicating the number of copies of a particular license, and obtain master licenses with an id that is reserved (O.ID and OE.MASTER).

T.LICENSE deals with obtaining confidential data in a license and is met through O.LICENSE. During transfer this is met by O.TRANSPORT. The hardware must be tamper resistant (OE.INSTALL) and detect integrity errors (O.STRESS).

T.CONFIDENTIAL considers the possibility to obtain security sensitive data in a license and is directly met by O.CONFIDENTIAL. A secure license transfer must also exist (O.TRANSPORT), as well as a tamper resistant hardware platform (OE.INSTALL). This objective is further supported by the security by the vendor (OE.APPLIC).

T.TIME identifies a specific subset of the threat related to time limited licenses. This specific threat is addressed by O.TIMELIMIT. This objective is also supported by the execution of QX applications (O.EXECUTE).

T.NUMBER identifies a specific subset of the threat related to execution of software more times than permitted by a license. This specific threat is addressed by O.NUMLIMIT. The increase of the counter in a license is specified by the vendor and is included in the protected code. This code is protected from modification (O.INTEGRITY). This objective is also supported by the execution of QX applications (O.EXECUTE). Further, a QX application resident on a token must not be tampered with (OE.INSTALL).

T.ACR identifies a specific subset of the threat related to execution of software without a correct ACR field in a license (O.ACR). As the ACR is included in the QX applications, these strings must be protected from modification (O.INTEGRITY). This objective is also supported by the execution of QX applications (O.EXECUTE). Further, the QX application resident on a token must not be tampered with (OE.INSTALL).

T.INTERFERE is a threat dealing with the specific issue of information leaking between multiple applications, and is addressed by O.NONINTERFERE.

T.SECURE is a threat to the TOE concerned with hardware attacks on a token. This is addressed by the environmental objective OE.INSTALL.

T.MASTER is addressed by O.ID and OE.MASTER which controls the ability to obtain master licenses. Further the request for more master licenses of a given id is integrity protected (O.TRANSPORT). The transfer of master licenses from the master license production facility to the destination token is also addressed by O.TRANSPORT. The vendor must also protect their master licenses (OE.APPLIC). It shall not be possible to modify other licenses to a particular master license (O.LICENSE. OE.INSTALL) by changing the license identification number.

T.USE is the specific threat of an unauthorized person gaining unauthorized access to a token containing a valid license. There is an objective to control access to licenses on a token or the entire token, based on the status of PIN/PUK or passwords given by a user (O.USE). It is also possible that such an unauthorized person may attempt physical attacks on the token (OE.INSTALL) or read lock parameters of licenses (O.CONFIDENTIAL), either when a license is present on a token or in transfer. The lock parameters are security sensitive and must be protected (O.TRANSPORT). OE.SRS is used for giving a metric for the security of the lock functionality.

T.STRESS identifies the threat of an attacker modifying certain bits in a license by using write cycle limitations of EEPROM. This is met by the objective O.STRESS.

11.3 Security Requirements Rationale

11.3.1 Requirements Are Appropriate

The following table identifies which SFRs satisfy the objectives defined in section 7.1.1.

Security Functional Requirements	O.EXECUTE	O.CONFIDENTIAL	O.INTEGRITY	O.NONINTERFERE	O.PROTECT	O. TIMELIMIT	O.NUMLIMIT	O.ACR	O.TRANSPORT	O.COPY	O.USE	O.LICENSE	O.STRESS	O.ID	OE.INSTALL	OE.MASTER	OE.SRS
FCS_CKM.1									✓								
FCS_COP.1 (1)	✓		✓		✓		✓	✓									
FCS_COP.1 (2)		✓							✓	✓	✓	✓		✓			
FCS_COP.1 (3)									✓	✓	✓	✓		✓			
FDP_ACC.1 (1)		✓				✓	✓	✓	✓	✓	✓	✓					✓
FDP_ACC.1 (2)	✓				✓	✓	✓	✓									
FDP_ACF.1 (1)		✓				✓	✓	✓	✓	✓	✓	✓					✓

Security Functional Requirements	O.EXECUTE	O.CONFIDENTIAL	O.INTEGRITY	O.NONINTERFERE	O.PROTECT	O. TIMELIMIT	O.NUMLIMIT	O.ACR	O.TRANSPORT	O.COPY	O.USE	O.LICENSE	O.STRESS	O.ID	OE.INSTALL	OE.MASTER	OE.SRS
FDP_ACF.1 (2)	✓				✓	✓	✓	✓									
FDP_DAU.1														✓			
FDP_ITC.2 (1)		✓								✓							
FDP_ITC.2 (2)	✓						✓	✓									
FDP_RIP.1	✓	✓		✓	✓												
FDP_SDI.1			✓									✓	✓				
FDP_UCT.1 (1)		✓							✓	✓	✓	✓					
FDP_UCT.1 (2)			✓		✓												
FDP_UIT.1 (1)									✓	✓		✓		✓			
FDP_UIT.1 (2)			✓														
FIA_AFL.1 (1)											✓						
FIA_AFL.1 (2)											✓						
FIA_AFL.1 (3)											✓						
FIA_UAU.1											✓						
FIA_UAU.6											✓						
FPT_RPL.1										✓							
FPT_RVM.1	✓		✓			✓	✓	✓			✓						
FPT_SEP.1		✓		✓	✓												
FDP_DAU.1																✓	
FIA_SOS.1																	✓
FPT_PHP.1															✓		
FPT_PHP.3		✓													✓		
FDP_STM.1						✓											

Table 11-2 Mapping of objectives to SFRs

O.EXECUTE is met through the implementation of the QX application access control policy (FDP_ACC.1 (2) and FDP_ACF.1 (2)). FDP_ITC.2 (2) ensures that imported QX applications are correctly associated with

its security parameters. Memory locations used by other QX applications are cleared before reuse (FDP_RIP.1). Encryption is used to enforce the confidentiality of a QX application not present on a token (FCS_COP.1 (1)). Use of FPT_RVM.1 ensures that the policy is always upheld.

O.CONFIDENTIAL is met through the implementation of the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)) and the data exchange confidentiality (FDP_ITC.2 (1) and FDP_UCT.1 (1)). This is supported through the use of encryption (FCS_COP.1 (2)), and separation (FPT_SEP.1) and clearance of memory areas (FDP_RIP.1). The confidentiality depends also on the tamper resistant hardware platform (FPT_PHT.3).

O.INTEGRITY integrity protects QX applications for modification. This is met through the implementation of the data exchange integrity protection (FDP_UIT.1 (2)). This mechanism uses encryption for confidentiality protection (FCS_COP.1 (1) and FDP_UCT.1 (2)). When the QX applications are stored on tokens, they are monitored for integrity changes (FDP_SDI.1). Use of FPT_RVM.1 ensures that the policy is always upheld.

O.NONINTERFERE is met through the implementation of domain separation (FPT_SEP.1) and subset residual information protection (FDP_RIP.1) by requiring that memory for one application is erased before allocated to another application. Thus the interference does not require concurrently execution of QX applications. This ensures that no QX applications can interfere either with other QX applications or licenses.

O.PROTECT is met through the implementation of the QX application access control policy (FDP_ACC.1 (2) and FDP_ACF.1 (2)), and through use of encryption (FCS_COP.1 (1) and FDP_UCT.1 (2)) and separation (FPT_SEP.1) of QX applications. Memory allocation (FDP_RIP.1) ensures that memory location used by one QX application is cleared before another QX application uses the same location, thus it is not possible to get one QX application to read data used by a previous QX application.

O.TIMELIMIT is met through the implementation of the QX application access control policy (FDP_ACC.1 (2) and FDP_ACF.1 (2)) and the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). This mechanism requires reliable time stamps from the host (FDP_STM.1). Use of FPT_RVM.1 ensures that the time constraint is always checked before a QX application is executed.

O.NUMLIMIT is met through the implementation of both the QX application access control policy (FDP_ACC.1 (2) and FDP_ACF.1 (2)) and the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). The information to use a counter constraint is given in the QX application (FDP_ITC.2 (2)) as part of encrypted code (FCS_COP.1 (1)). Use of FPT_RVM.1 ensures that the number constraint of a license is always checked before a QX application or a part of this, is executed.

O.ACR is met through the implementation of the QX application access control policy (FDP_ACC.1 (2) and FDP_ACF.1 (2)) and the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). The ACR field is included in the protected QX application as a security attribute (FDP_ITC.2 (2)), thus part of encrypted code (FCS_COP.1 (1)). Before execution of a QX application, the ACR is checked (FPT_RVM.1).

O.TRANSPORT is met through the implementation of the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). This is supported through use of encryption and cryptographic hashing (FCS_COP.1 (2) and FCS_COP.1 (3)). In transport, a session key is used (FCS_CKM.1). The transport mechanism implements an atomic operation consisting of countdown of copies of the license available and a

generation of a transport record (, and any interference during transmission can be detected (FDP_UIT.1 (1)). The confidentiality is ensured by FDP_UCT.1 (1).

O.COPY is met through the implementation of the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). The mechanisms for preventing double spending or replay attacks use cryptographic means (FCS_COP.1 (2) and FCS_COP.1 (3)) for confidentiality (FDP_UCT.1 (1)) and integrity (FDP_UIT.1 (1)) protection of these parameters during transfer of a license. When the license is uploaded to the destination token (FDP_ITC.2 (1)), means for detecting double spending or replay are provided (FPT_RPL.1).

O.USE is met through the implementation of the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). To use a license or token, either for execution of QX applications or specific license operations which can be locked, the user must provide a pin or password in order to get authorized (FIA_UAU.1). A user must re-authenticate if the token is re-connected (FIA_UAU.6). After a specific number of unsuccessful authorizations, the lock is blocked (FIA_AFL.1 (1), FIA_AFL.1 (2) and FIA_AFL.1 (3)). As the lock attributes are parts of licenses, they are also confidentiality and integrity protected during transfer (FCS_COP.1 (2), FCS_COP.1 (3) and FDP_UCT.1 (1)). There is no bypassability of this policy (FPT_RVM.1).

O.LICENSE controls the rights to modify license attributes and is met be the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)). When the licenses are stored on tokens, they are monitored for integrity changes (FDP_SDI.1), and outside control of the TOE, the message digest of the license (FCS_COP.1 (3)) is encrypted (FCS_COP.1 (2) and FDP_UCT.1 (1)) for detection of integrity modifications (FDP_UIT.1 (1)).

O.STRESS is met by the detection of write failures leading to integrity errors of data stored on tokens (FDP_SDI.1).

O.ID governs the right to obtain more copies of an already allocated license id from a master license production facility. FDP_DAU.1 ensures that such authentication can be provided. Further, this information is integrity protected when sent to the master license production facility (FCS_COP.1 (2), FCS_COP.1 (3) and FDP_UIT.1 (1)).

OE.INSTALL is met by FPT_PHP.1 and FPT_PHP.3, which require detection and prevention of hardware attacks.

OE.SRS is met through the implementation of the license access control policy (FDP_ACC.1 (1) and FDP_ACF.1 (1)) by providing a metric for the lock functionality (FIA_SOS.1).

OE.MASTER ensures that a master license production facility has mechanisms for verifying a proof of ownership of a master license with a particular identification number (FDP_DAU.1).

11.3.2 Security Requirement Dependencies Are Satisfied

Functional Component	Dependencies	SFR(s) in Security Target meeting Dependencies
FCS_CKM.1	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4 FMT_MSA.2	FCS_COP.1 (FCS_CKM.4 is not used since all keys will be used within the tamper resistant environment of the TOE. FMT_MSA.2 is implemented by the Sospita Runtime System [SRS][13]).
FCS_COP.1	[FDP_ITC.1 or FCS_CKM.1] FCS_CKM.4 FMT_MSA.2	FCS_CKM.1 (FCS_CKM.4 is not used since all keys will be used within the tamper resistant environment of the TOE. FMT_MSA.2 is implemented by the Sospita Runtime System [SRS][13]).
FDP_ACC.1	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1 (FMT_MSA.3 is not used, since policy attributes are set externally, and secure default values are not used.)
FDP_DAU.1	-	-
FDP_ITC.2	[FDP_ACC.1 or FDP_IFC.1] [FTP_ITC.1 or FTP_TRP.1] FPT_TDC.1	FDP_ACC.1 (FDP_UIT.1 and FCS_COP.1 replace FTP_ITC.1 as integrity, rather than confidentiality is the principal concern. FPT_TDC.1 is omitted as all exchanges are with other parts of the Sospita product, and the need to supply data in particular formats is not present.
FDP_RIP.1	-	-
FDP_SDI.1	-	-
FDP_UCT.1	[FTP_ITC.1 or FTP_TRP.1] [FDP_ACC.1 or FDP_IFC.1]	FDP_ACC.1 (FTP_ITC.1 or FTP_TRP.1 is not used since protection of applications should be performed in a trusted computer environment. Secure user attributes apply only to end-users, and end-user are self-responsible for their computing environment).
FDP_UIT.1	[FDP_ACC.1 or FDP_IFC.1] [FTP_ITC.1 or FTP_TRP.1]	FDP_ACC.1 (FTP_ITC.1 or FTP_TRP.1 is not used since protection of applications should be performed in a trusted computer environment. Secure user attributes apply only to end-users, and end-user are self-responsible for their computing environment).
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1
FIA_UAU.1	FIA_UID.1	(FIA_UID.1 is not used since the token itself is the

[13] This applies only to weak and semi-weak encryption keys.

Functional Component	Dependencies	SFR(s) in Security Target meeting Dependencies
		identification of the user).
FIA_UAU.6	-	-
FPT_RPL.1	-	-
FPT_RVM.1	-	-
FPT_SEP.1	-	-

Table 11-3 Mapping of SFR dependencies

11.3.3 Security Requirements Are Mutually Supportive

The only interactions between the security requirements specified for the TOE are those which are identified in the CC Part 2 as dependencies between the SFRs. These dependencies are documented and demonstrated to be satisfied in section 11.3.2. These interactions are specified in the CC Part 2, and are therefore mutually supportive.

11.3.4 ST Complies With the Referenced PPs

This Security Target does not claim compliance with a Protection Profile.

11.4 IT Security Functions Rationale

11.4.1 IT Security Functions Are Appropriate

The Table below provides a mapping of chapter 9 IT functions to SFRs (Section 7.1).

IT Function	Security Functional Requirement(s)																							
	FCS_CKM.1	FCS_COP.1(1)	FCS_COP.1(2)	FCS_COP.1(3)	FDP_ACC.1(1)	FDP_ACC.1(2)	FDP_ACF.1(1)	FDP_ACF.1(2)	FDP_DAU.1	FDP_ITC.2 (1)	FDP_ITC.2 (2)	FDP_RIP.1	FDP_SDI.1	FDP_UCT.1 (1)	FDP_UCT.1 (2)	FDP_UIT.1 (1)	FDP_UIT.1 (2)	FIA_AFL.1 (1)	FIA_AFL.1 (2)	FIA_AFL.1 (3)	FIA_UAU.1	FIA_UAU.6	FPT_RPL.1	FPT_RVM.1	FPT_SEP.1
F1						✓		✓																✓	
F2					✓		✓																✓	✓	
F3					✓		✓																	✓	
F4			✓		✓		✓			✓				✓										✓	
F5			✓	✓										✓		✓									
F6					✓		✓																✓	✓	
F7					✓		✓																✓	✓	
F8					✓		✓																✓	✓	
F9					✓		✓		✓															✓	
F10					✓		✓			✓													✓	✓	
F11			✓		✓		✓							✓											
F12					✓		✓																	✓	
F13						✓		✓																✓	
F14												✓													✓
F15						✓		✓			✓													✓	
F16			✓		✓		✓							✓										✓	
F17					✓		✓						✓											✓	
F18		✓				✓		✓							✓		✓								
F19	✓		✓		✓		✓			✓				✓										✓	
F20					✓		✓																	✓	
F21					✓		✓														✓	✓		✓	

IT Function	Security Functional Requirement(s)																							
	FCS_CKM.1	FCS_COP.1(1)	FCS_COP.1(2)	FCS_COP.1(3)	FDP_ACC.1(1)	FDP_ACC.1(2)	FDP_ACF.1(1)	FDP_ACF.1(2)	FDP_DAU.1	FDP_ITC.2 (1)	FDP_ITC.2 (2)	FDP_RIP.1	FDP_SDI.1	FDP_UCT.1 (1)	FDP_UCT.1 (2)	FDP_UIT.1 (1)	FDP_UIT.1 (2)	FIA_AFL.1 (1)	FIA_AFL.1 (2)	FIA_AFL.1 (3)	FIA_UAU.1	FIA_UAU.6	FPT_RPL.1	FPT_RVM.1	FPT_SEP.1
F22																		✓	✓					✓	
F23																				✓				✓	
F24					✓		✓																	✓	
F25													✓												
F26													✓												
F27					✓		✓																	✓	

Table 11-4 Mapping of IT functions to SFRs

FCS_CKM.1 is met by the use of session keys (F19).

FCS_COP.1(1) is met by the encryption of QX applications in function F18.

FCS_COP.1(2) describes encryption of licenses. This happens when a license leaves a token during transport and backup (F4) . A hash of the transport record is encrypted for ensuring the integrity (F5). F16 describes the encrypted attributes for a license outside a token, and this is further covered by F11. Encryption of session key is also performed (F19).

FCS_COP.1(3) considers the generation of a message digest. This applies to licenses being transferred (F5).

FDP_ACC.1(1) is met by several functions considering the functions that can be performed on licenses (F2, F3, F4, F6, F7, F8, F9, F10, F11, F12, F16, F17, F19, F20, F21, F24 and F27).

FDP_ACC.1(2) is met by functions considering the functions that can be performed on QX applications (F15), and the qualities of QX applications (F1 and F13).

FDP_ACF.1(1) is a refinement of FDP_ACC.1(1) and is met by several functions considering the functions that can be performed on licenses (F2, F3, F4, F6, F7, F8, F9, F10, F11, F12, F16, F17, F19, F20, F21, F24 and F27).

FDP_ACF.1(2) is a refinement of FDP_ACC.1(2) and is met by functions considering the functions that can be performed on QX applications (F15), and the qualities of QX applications (F1 and F13).

FDP_DAU.1 is met by F9 by using proof of allocated license id.

FDP_ITC.2(1) is met by functions which require import of licenses (F4 and F10). A session key is also a security parameter (F19).

FDP_ITC.2(2) is met by function which require upload of QX applications (F15).

FDP_RIP.1 describes the memory locations which are cleared before re-use. This is met by F14.

FDP_SDI.1 considers the integrity monitoring of licenses and QX applications stored on tokens. This is met by F17. Uncontrollable hardware fault is covered by F25 and F26.

FDP_UCT.1(1) is met by the functions describing the confidentiality protection of licenses (F11 and F16). This mechanism is invoked when a license is not on a token (F4). The additional encrypted attributes of a transport record are the message digest (F5) and the session key (F19).

FDP_UCT.1(2) is met by the function describing the confidentiality protection of QX applications (F18).

FDP_UIT.1(1) is met by the function describing the integrity protection of licenses (F5).

FDP_UIT.1(2) is met by the functions describing the integrity protection of QX applications (F18).

FIA_AFL.1(1) is met through F22 by allowing three attempts to give a correct PIN before a PIN-locked license or token can be unlocked by the use of a PUK.

FIA_AFL.1(2) is met through F22 by allowing six attempts to give a correct PUK before a PIN-locked license or token is permanently un-lockable.

FIA_AFL.1(3) is met through F23 by allowing ten attempts to give a correct password before a password-locked license or token is permanently un-lockable.

FIA_UAU.1 is met through F21, by requiring the user to authenticate before using a license or token.

FIA_UAU.6 is met through F21, by requiring the user to re-authenticate when a token is re-connected to the host.

FPT_RPL.1 is met by the detection of replay or double spending. This is directly met by F7 and F8 during transfer. Illegal copying of licenses is further controlled by the license instance counter (F2, F6 and F10).

FPT_RVM.1 considers the non-bypassability of the TSP and is met by several functions (F1, F2, F3, F4, F6, F7, F8, F9, F10, F12, F13, F15, F16, F17, F19, F20, F21, F22, F23, F24 and F27).

FPT_SEP.1 is met by F14 which considers the firewall mechanism between QX applications an licenses.

As can be seen by the above table and rationale, all Security Functional Requirements of the TOE are fully provided by the IT security functions specified in the TOE Summary Specification.

Also demonstrated in Table 11-4, all IT Security Functions identified for the TOE in the TOE Summary Specification are required to meet the TOE Security Functional Requirements.

11.4.2 IT Security Functions Are Mutually Supportive

The mutually supportive nature of the IT security functions can be derived from the mutual support of the SFRs (demonstrated in Section 11.3.3), as each of the IT functions can be mapped to one or more SFRs, as demonstrated in Table 11-4.

11.4.3 Strength of Function Claims Are Appropriate

The SoF claim made by the TOE is SOF-HIGH, which is defined in the CC Part 1 as:

A level of the TOE strength of function where analysis shows that the function provides adequate protection against deliberately planned or organised breach of TOE security by attackers possessing a high attack potential.

AVA_VLA.1, one of the assurance components from which the EAL3 assurance level is comprised, which determines that obvious vulnerabilities cannot be exploited in the intended environment of the TOE (CC Part 3). Therefore, a SoF claim of SOF-HIGH demonstrates that the functions with an associated SoF would ensure that obvious vulnerabilities have been addressed.

Therefore, the claim of SOF-HIGH made by The TOE is viewed to be appropriate for this use.

11.4.4 Assurance Measures Satisfy Assurance Requirements

Table 11-5, below, provides a tracing of the Assurance Measures to the assurance requirements that they meet. From the table it can be seen that all assurance requirements trace to at least one assurance measure.

The assurance requirements identified in the table are those required to meet the CC assurance level EAL3. As all assurance requirements are traced to at least one of the assurance measures, the identified assurance measures are sufficient to meet the assurance requirements. It is also asserted that the assurance measures have been produced with EAL3 in mind and as a consequence contains sufficient information to meet the assurance requirements of the TOE.

Assurance Measures (documentation)	Assurance Requirements Met by Assurance Measure	
Configuration Management Documentation	ACM_CAP.3	Authorization Controls
	ACM_SCP.1	TOE CM coverage
Delivery, Installation and generation Procedures	ADO_DEL.1	Delivery Procedures
	ADO_IGS.1	Installation, generation and start-up procedures
Development Environment Security Measures	ALC_DVS.1	Identification of Security Measures
Functional Specification Documentation	ADV_FSP.1	Informal Functional Specification
High-Level Design Documentation	ADV_HLD.2	Security enforcing high-level design
Correspondence Demonstration Document	ADV_RCR.1	Informal correspondence demonstration
Administrator Guidance Documentation	AGD_ADM.1	Administrator guidance
User Guidance Documentation	AGD_USR.1	User guidance
Test Coverage and Depth Documentation	ATE_COV.2	Analysis of coverage
	ATE_DPT.1	Testing: high-level design
Test Plan and actual tests and Results	ATE_FUN.1	Functional testing

Assurance Measures (documentation)	Assurance Requirements Met by Assurance Measure	
Independent Testing Resources	ATE_IND.2	Independent testing
Administrator and User Guide	AVA_MSU.1	Examination of Guidance
Strength of Function Documentation	AVA_SOF.1	Strength of TOE security function evaluation
Vulnerability Assessment Analysis Documentation	AVA_VLA.1	Developer vulnerability analysis

Table 11-5 Mapping of assurance measures to assurance requirements

Below, the documents for the different assurance requirements are listed. This list is not included in the public version of the QX Security Target.

12 Document history

Date	Description
24 May 2002	Draft version created
20 June 2002	Final draft for review by System Sikkerhet
1 July 2002	Modified the TOE introduction
5 July 2002	Minor modifications
15 July 2002	Changes in the template
8 August 2002	Updated with comments from SERTIT (section 5.3)
11 September 2002	Public version created